Introduction to the Geant4 Simulation toolkit

S. Guatelli[a], D. Cutajar[a], B. Oborn[b,a], A. B. Rosenfeld[a]

[a]*Centre for Medical Radiation Physics (CMRP), University of Wollongong, Northfields Avenue, Wollongong, NSW, 2522, Australia*
[b]*Illawarra Cancer Care Centre (ICCC), Wollongong Hospital, Crown Street, Wollongong, NSW, 2500, Australia*

Abstract. Geant4 is a Monte Carlo simulation Toolkit, describing the interactions of particles with matter. Geant4 is widely used in radiation physics research, from High Energy Physics, to medical physics and space science, thanks to its sophisticated physics component, coupled with advanced functionality in geometry description. Geant4 is widely used at the Centre for Medical Radiation Physics (CMRP), at the University of Wollongong, to characterise and optimise novel detector concepts, radiotherapy treatments, and imaging solutions. This lecture consists of an introduction to Monte Carlo method, and to Geant4. Particular attention will be devoted to the Geant4 physics component, and to the physics models describing electromagnetic and hadronic physics interactions. The second part of the lecture will be focused on the methodology to adopt to develop a Geant4 simulation application.

Keywords: Monte Carlo method, Geant4, Geant4 application development.
PACS: 87.55.K, 87.55.kh

INTRODUCTION

This lecture is an introduction to the Geant4 Simulation Toolkit [1, 2], addressed to Geant4 beginners and this lecture note summarizes various information previously provided by the Geant4 Collaboration.

Geant4 is widely used in the world and at the Centre for Medical Radiation Physics (CMRP), of the University of Wollongong, as a Monte Carlo Toolkit for radiation physics, from High Energy Physics (HEP) to medical physics and space science.

In particular, at the CMRP, Geant4 is widely used to characterise novel detector concepts such as Silicon-On-Insulator microdosimetry [3-10], fast neutron Medipix-based dosimeters for radiation protection in aviation and space [11], MOSFET characterisation [12], etc. Geant4 is also used to study possible novel radiotherapy treatments. For example, [13] describes the possibility to enhance the effectiveness of radiotherapy, thanks to the introduction of a magnetic field in the treatment region. At CMRP Geant4 is widely used also for protontherapy related studies as the one described in [14].

This lecture is structured in three parts. The first one describes the Monte Carlo method, the second one the functionality of the Toolkit, with particular attention devoted to the Geant4 physics components and its validation, as this is the core of the Monte Carlo code. The third part describes the main components of a Geant4 simulation application, and the methodology to adopt to develop a new Geant4 application.

To assist further beginners, a Geant4 simple application, the `basic_brachy`, has been provided on the USB pen drive, to adopt as a starting point to develop a specific Geant4 application. The student may also want to retrieve the concepts explained in this lecture in the Geant4 application provided.

We suggest that students to solve the Geant4 exercises proposed at the end of this paper, to start to learn to use Geant4, as the best way to learn a Monte Carlo code is to use it with the support of documentation. The exercises support the student to learn to develop a dosimetric brachytherapic system, adding new functionality to the `basic_brachy` application, adopting an iterative-incremental approach. After this work, the students will be able to develop any Geant4 simulation application.

This lecture provides an introduction to Geant4 and to the basic ingredients to develop a Geant4 simulation. To get an in-depth knowledge of the Toolkit, it is highly recommended to read at least the major part of the bibliographic references, suggested in this paper.

INTRODUCTION TO MONTE CARLO

Monte Carlo is the term given to the method of solving problems using random events. The term was coined in the 1940s by Stanislaw Ulam, Los Alamos National Laboratory, as he had an uncle who frequented the Casinos of Monte Carlo. Whilst working on solving the problems of nuclear diffusion in fissionable materials, a computer program based on random numbers was constructed and executed on the first electronic computer, ENIAC[15]. The concept of solving problems using random events, however, has been around for centuries. Random events may be generated in many ways other than through the use of a computer algorithm. Historically, calculations based on random events have been performed long before the development of computers. In the 18th century, Georges Louis Leclerc, Comte de Buffon, proposed an experiment involving the dropping of a needle onto a surface containing equidistant parallel lines. This experiment is known as Buffon's needle. The probability that the needle will cross any of the parallel lines is a function of the needle length, the line separation and π, according to the equation

$$P = \frac{2L}{t \cdot \pi}$$

Where P is the probability of the needle crossing any of the parallel lines, L is the length of the needle and t is the line separation. In the 19th century, Pascal proposed the estimation of π using Buffon's needle. If the needle is dropped a large number of times then π may be estimated using the equation

$$\pi \approx \frac{2L \cdot n}{t \cdot h}$$

Where n is the number of times the needle is dropped and h is the number of times the needle crosses one of the parallel lines. In 1901, Mario Lazzarini performed Buffon's needle experiment, yielding the approximation of π to be 355/113, which estimates π to within 10^{-6} of currently accepted values [16].

The concept of Monte Carlo calculation has been employed in recent times in the field of radiation transport due to its ability to model very complex systems. There are many Monte Carlo codes that have been developed for particle transport, including GEANT4, EGS [17], PENELOPE [18] and MCNP [19]. Despite being unique, all Monte Carlo radiation transport codes follow similar processes of calculation:

1. Define a set of possible inputs and restraints for a system.
2. Randomly generate inputs into the system and perform a computation.
3. Aggregate the results of the individual computations into the final result.

Define a Set of Possible Inputs and Restraints

The first step in the Monte Carlo process involves the definition of the requirements to be satisfied by the system to be modeled. For radiation transport Monte Carlo codes, these include:

- Experimental set-up, in terms of material composition and geometry and detectors.
- Definition of the primary radiation field (type, energy, direction, number of particles).
- Physics interactions.
- Result of the calculation (dose, kerma, spectra, etc.).
- Variance reduction techniques.
- Etc.

Randomly Generate Inputs Into the System and Perform a Computation

The second step in the Monte Carlo process involves the generation of random events within the defined system. Most Monte Carlo codes for radiation transport follow the same routines for random particle generation and tracking, which is as follows:

- The path of a primary particle with the predetermined initial conditions is simulated through the initial medium
- The possible interactions of the primary particle with the medium are determined and the probability of each interaction calculated
- A random number generator is used to randomly select an interaction
- The amount of energy lost due to the interaction is recorded and tabulated for that region in the medium

- Any secondary particles created from the interaction are then tracked and each subsequent interaction recorded and energy loss tabulated
- The primary particle and all other secondary particles are tracked until the energy of each particle is below the cutoff energy, at which point the energy is said to have been lost at this point and the energy loss tabulated
- Another primary particle is generated and tracked
- This process is repeated until the predetermined amount of primary particles to be simulated has been reached

Aggregate the Results of the Individual Computations into the Final Result

The third and final step in the Monte Carlo process involves the final analysis of the obtained data. For radiation transport Monte Carlo codes, this may involve the calculation of dosimetric quantities, such as the dose deposited per generated primary particle within a defined region. The final result or results may be placed in any form representative of the calculated result as per the wishes of the user. Such forms include individual point doses, 1, 2 or 3 dimensional dose maps and energy spectra.

The Monte Carlo method is a very powerful and useful tool as it allows for the solving of complex problems which may not have analytical solutions. With the recent advancement in computing speeds, the Monte Carlo method for dosimetric applications is gaining popularity, and will hopefully be a common tool for clinical dose planning in the near future.

THE GEANT4 TOOLKIT

Introduction

Geant4 [1, 2] is a Monte Carlo simulation Toolkit, modeling the interactions of particles with matter. It was born in the early 90s, at CERN (Geneva, Switzerland), as the RD44 project. The goal was to develop a robust and versatile simulation software tool, based on Object-Oriented technology [1], capable of satisfying the requirements imposed by the modern particle and nuclear physics, dictated by the particle detector complexity of the large-scale High Energy Physics (HEP) experiments (i.e. the Large Hadron Collider, at CERN).

The R&D phase was completed in December 1998 [20], with the delivery of the first production release. Subsequently the Geant4 Collaboration was established in January 1999 to continue developing and refining the Toolkit. The Collaboration also provides maintenance of the Toolkit and user support to those using Geant4 [1].

From that time, the Geant4 Collaboration, counting nowadays more than 100 collaborators, from various research institutes all around the world have not stopped developing the Toolkit. Each year two public releases of Geant4 are provided, with new and/or refined functionality.

Geant4 was born as a powerful Monte Carlo simulation Toolkit for HEP. However, as a result of its flexibility and openness to evolution, it was extended to other physics research domains, as space science [21] and medical physics [22].

Nowadays Geant4 is the only open-source, free and general Monte Carlo code for radiation physics research, from High Energy Physics [23] down to eV scale [24, 25]. Geant4 provides comprehensive detector and physics modeling capabilities. It is possible to model the experimental set-up in terms of geometry and materials and to define the particles involved and their physics interactions. The user can track particles in matter, also in the presence of electromagnetic fields, and describe the response of the detector. Geant4 provides interfaces to enable its users to interact with their simulation application and store their results in analysis objects (histograms, ntuples, etc.). Visualisation drivers and graphical user interfaces are included in the Toolkit.

The main advantage of using Geant4 as a Monte Carlo code, determining its success in radiation physics research, is provided by its wide offer of complementary and alternative physics models describing both the electromagnetic and hadronic interactions of particles with matter.

In the next sub-sections the Geant4 kernel will be introduced in its main features, followed by the description of important aspects of the Geant4 components such as *detector*, *particle*, *tracking* and *physics*.

Note that the next sub-sections are intended to be an introduction to some fundamental Geant4 functionalities, as a first approach to studying the Monte Carlo Toolkit. It is highly recommended to refer to the Geant4 Documentation [26] for in-depth studies.

The Geant4 kernel

Fig.1 illustrates the modular, hierarchical structure of the Geant4 kernel. Each category manages a component of the software. The sub-domains are linked by a uni-directional flow of dependencies.

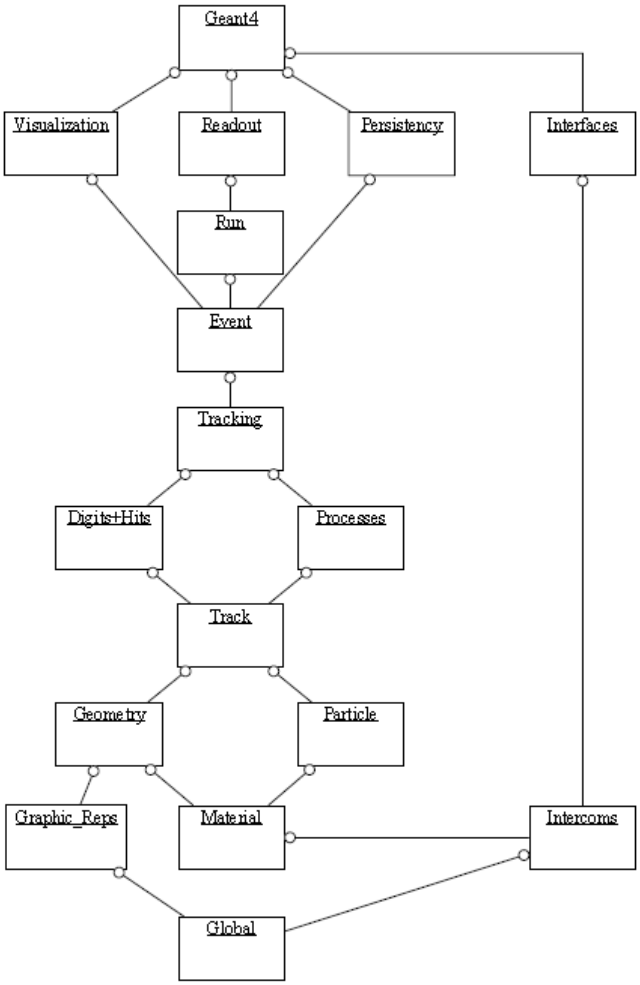

FIGURE 1. The Top Level Category Diagram of the Geant4 Toolkit. The open circle on the joining lines represents a using relationship; the category at the circle end uses the adjoined category [1]. Picture courtesy of Geant4 Collaboration.

The categories (packages) at the bottom of the diagram are used by higher categories and provide the foundation of the Toolkit. These include the package *global* (covering the system of units, constants, numerics and random number handling), *materials*, *particles*, *graphical representations*, *geometry* (including the volumes for detector description and the particle navigation in the geometry model), and *intercoms*, which provides both a means of interacting with Geant4 through the user interface and also a way of communicating between modules that should not otherwise depend on one another.

Above these reside packages required to describe the tracking of particles and the physical processes they undergo. The *track* package contains classes for tracks and steps, used by *processes*, which contain implementations of models of physical interactions. Additionally, one such process, *transportation*, handles the transport of particles in the geometry model. All these processes may be invoked by the *tracking* package, which manages their contribution to the evolution of a track's state and provides information from hits in the sensitive volumes (i.e. energy deposition, generation of secondary particles, etc.), and digitisation (model of the detector response).

Over these the *event* package manages events and *run* manages collections of events that share a common experimental configuration. A *readout* package allows handling the description of the electronics and the readout associated to the experimental set-up.

Capabilities that use all of the above and connect to facilities outside the toolkit are provided by the *visualisation*, *persistency* and *user interface* packages.

The next sub-sections describe in more detail the main features of the most important packages of Geant4, in order to allow the user to develop a first Geant4 application.

Geometry and materials

The *geometry* package offers the ability to describe a geometrical structure and propagate particles through it. A component of the experimental set-up is defined geometrically, through three layers:

- *Solid*: the shape and the size of a geometrical component are defined.
- *Logical volume*: the attributes (material, sensitivity, presence of electromagnetic fields, etc.) of the component are defined.
- *Physical volume*: definition of the spatial positioning or placement of the logical volume with respect to the enclosing mother volume.

The experimental set-up of the Geant4 simulation is modelled with a hierarchical tree structure, each volume containing smaller volumes. The user has to model the experimental set-up of his/her own Geant4 simulation, hierarchically, starting from top level geometries (Box 1 of Fig. 2).

Fig. 2 shows an example of a Geant4 application experimental set-up that models three boxes. Box1 is the biggest box, containing Box2, containing Box3. In Geant4 language, this means that Box1 is the mother volume of Box2. Analogously Box2 is mother volume of Box3.

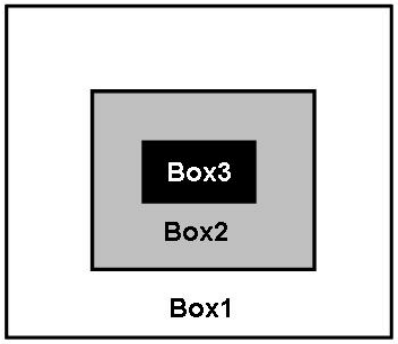

FIGURE 2. Simple experimental set-up of a Geant4 dummy simulation application. Box1 is mother volume of Box2, which is mother of volume Box3. Box1 contains all the geometrical components of the experimental set-up; it is the so-called *World* volume.

Box1 is the volume containing all the experimental set-up of the Geant4 simulation, its mother volume is 0, and is usually called *World* volume.

In a Geant4 experimental set-up, volumes must be entirely contained in their mothers. There cannot be intersections between volumes of the same or different hierarchical level. If there are, these are software bugs and must be corrected.

The *materials* category reflects what exist in nature. Materials are made of single element or a mixture of elements. Similarly elements are made of single isotope or a mixture of isotopes. A single element material, such as liquid argon, can be defined as shown in Fig. 3. Fig. 4 shows how a compound, such as water, can be modelled.

```
G4double density = 1.390* g/cm3;
G4double a = 39.95*g/mole;
G4double z = 18. ;
G4Material* liquidArgon = new G4Material
                    ("liquidArgon", 18., a, density);
```

FIGURE 3. Definition of a single element material, in Geant4.

```
a = 1.01*g/mole;
G4Element* elH = new G4Element("Hydrogen",symbol="H",z=1.,a);

a = 16.00*g/mole;
G4Element* elO = new G4Element("Oxygen",symbol="O",z=8.,a);

density = 1.000*g/cm3;
G4Material* H2O = new G4Material("Water",density,ncomp=2);

G4int natoms;
H2O -> AddElement(elH, natoms = 2);
H2O -> AddElement(elO, natoms = 1);
```

FIGURE 4. Definition of water, in Geant4.

Debugging tools for geometry

One of the most common problems for Geant4 beginners is to correctly model the experimental set-up of the simulation application, and one of the most common errors is to model geometrical components that overlap one another. This happens when a volume extends beyond the boundaries of its mother volume, or when volumes are positioned within the same mother volume such that they intersect one another.

To support the user in avoiding the introduction of geometrical bugs, Geant4 offers various debugging tools, to verify the correct modelling of the experimental set-up. The strategy suggested is as follows. The user should model the experimental set-up of his/her specific use case incrementally, from the simpler geometry components to the more complex ones, adding and testing the geometrical refinements at each development cycle (for more details on the methodology to adopt to develop a software, read session "Software Process").

The testing procedure should include at least the following steps: (1) visualisation of the experimental set-up to gain an impression of the geometry being modelled; (2) use of the method *CheckOverlaps()* at the level of the physical volumes of the geometry, to verify possible geometrical volume overlaps, during the initialization phase of the simulation; (3) in case there is a geometrical overlap, use the debugging tool DAVID (DAWN's Visual Intersection Debugger) [27], to graphically highlight the overlapping regions of the experimental set-up. The last action is to correct the geometrical model and repeat the testing procedure.

The detector component

The *Detector* package manages the detector components of the experimental set-up. The *Detector* is the geometrical component information is retrieved regarding the physical interactions of particles, such as the energy deposition, the particle momentum, time and position of the interaction. A Geant4 *Detector* is created when the sensitivity attribute of the associated geometrical component is set at the level of the logical volume. There can be more detectors within the same Geant4 experimental set-up.

The user can store the information collected in the detector in an object called *Hit*. The *Hit* is a snapshot of the physical interaction of a track in the sensitive region of the detector. Then the user can retrieve the *Hit* collection at the end of each *event*, or *run*, and store this information in an output file.

The primary particle component

The primary particle component in Geant4 manages the modelling of the radiation field of interest, providing the type, energy, vertex, and momentum of the particles. The user can generate primary particles by means of the *G4ParticleGun* [1] or *GeneralParticleSource* (GPS) [28].

The use of the *G4ParticleGun* is suggested when the radiation field to model is simple, as in the case, for example, of a monochromatic particle beam. However, if the radiation field to be modelled is more complex (i.e. the galactic cosmic rays), the use of the GPS is suggested. This package allows the specifications of the spectral, spatial and angular distribution of the primary particles, through simple interactive commands, to be specified in a macro file, input of the simulation.

The tracking

The *Tracking* package manages the tracks of particles in the experimental set-up. A track is the sequence of steps of a particle. The step is defined as the distance between two interaction points. At the level of the step it is possible to retrieve information about the particle, e.g. its kinetic energy at begin and at the end of the step, its energy deposition during the step (associated with the beginning of the step), the physics processes undertaken etc.

The *Tracking* package steers the invocation of processes of particles in the experimental set-up. Geant4 treats physics processes in a very generic way; tracking does not depend on the particle type or on the specific physics process, including particle transportation.

Event and run

The *event* is the main unit of the simulation. It starts when one or more primary particles are generated and it concludes when all the primary and secondary particles are fully tracked in the experimental set-up. A *run* is a collection of events that share the same experimental set-up conditions. The configuration of the geometry, the detectors and the physics processes cannot be altered during a *run*.

Interactivity and visualisation

Interactivity and visualisation span the related packages: *intercoms*, *interfaces,* and *visualisation* (see Fig.1).

User interaction is realised through the concept of a "session" and graphical and non-graphical concrete sessions are available in the *interface* category. The user can define interactive commands to type in interactive sessions during the simulation, which can alter the configuration of the simulation.

The definition and use of interactive commands to change parameters of the simulation is suggested, as it allows the development a general and flexible simulation application, which can be used in different use cases, performed by different users. This strategy increases, on one side, the efficiency of the work, as the same code can be used in multiple studies, and, on the other side, the quality of the work since the same simulation code would be validated by different users, in different experimental set-ups. This strategy also helps to maintain a clean and clear code.

The user is able to visualise the geometrical configuration of the experimental set-up and the tracks of the particles. Drivers for several graphic systems are offered (OpenGL, DAWN, WRML, WIRED, RayTracer, etc.)[1], which can be instantiated in parallel in a simulation.

By default, the tracks of the particles are coloured on the basis of their charge: neutral particles such as photons, are coloured in green, negatively and positively charged particles, such as electrons and positrons, have tracks coloured red and blue respectively. The visualisation attributes of the geometrical components of the experimental set-up are defined at the level of the logical volumes.

It is highly recommended not to use visualisation tools when executing simulations with high number of primary particles as this may cause a very high use of memory.

The physics component

Geant4 provides a wide variety of physics components for use in the simulation. Geant4 physics processes (*G4Process* class) describe how particles interact with materials. Geant4 provides seven major categories of processes: electromagnetic, hadronic, transportation, decay, photolepton_hadron, and parameterisation.

Geant4 offers a wide set of alternative and complementary electromagnetic and hadronic physics models. For a beginner it can be very difficult to identify the best set of physics models, for his/her use case. The suggested approach consists is comparing the results derived from the Geant4 simulation, adopting different model approaches, with respect to experimental measurements, to identify the best physics list to adopt. This approach was for example adopted in the study [29], to identify the best combination of physics models of Geant4 addressed to protontherapy dosimetric studies. This process should be accompanied by a literature review to study and evaluate the approach adopted by other Geant4 users, in the use case of interest.

Another obstacle for Geant4 beginners is to learn to instantiate physics models and processes. Especially the definition of hadronic physics models can be difficult, at first impact. The suggestion is to start adopting one of the suggested guessed physics lists provided by Geant4, or by adopting a physics list of a Geant4 advanced example, with similar use case.

In the next sub-sections, details on the physics category and physics models will be provided. For more details please consult the Geant4 Application Developer Manual and the Geant4 Physics Reference Manual, available for free download from [26].

Physics processes

As known from radiation physics, a particle in flight is subject to many competing processes in a medium, with cross sections that depend on the type of particle, its energy, and the properties of the medium. In a real detector, the particle will often travel through many regions of different materials, shapes and sizes before interacting or decaying.

In Geant4, each particle is moved step by step in the simulation experimental set-up from the point of origin. A step, by default, is defined as the distance between two interaction points.

At the beginning of a step, all competing physics processes of the particle propose the next interaction point by sampling their interaction length (or mean free path). In a target with density ρ, made by *i* isotopes of mass m_i, with fraction x_i, the interaction length λ is calculated as follows:

$$\lambda = \frac{1}{\rho \sum_i \frac{x_i \sigma_i}{m_i}}$$

where σ is the cross section of the specific process.

The mean free path of a particle for a given process depends on the medium and cannot be used directly to sample the probability of an interaction in a heterogeneous detector.

From radiation physics we know that the probability of a particle to survive a distance dx is:

$$P(x) = e^{-\frac{dx}{\lambda}}.$$

The probability to survive along a path *l* is then:

$$P(l) = e^{-\int_0^l \frac{dx}{\lambda}},$$

where $\int_0^l \frac{dx}{\lambda(x)}$ is the number of mean free paths n_λ. Note that this physics quantity is independent from the material traversed and from the particle energy.

Now we can rewrite equation $P(l) = e^{-\int_0^l \frac{dx}{\lambda}}$ as $P(l) = e^{-n_\lambda}$. The key point to note is that the probability distribution of n_λ is a simple exponential independent of material and energy and can be used to sample the interaction point. So, we set:

$$n_\lambda = -\ln \eta,$$

where η is a random number uniformly distributed in the range (0, 1), and this is used to determine the distance *l* to the point of interaction in the current material, for the specific process.

The interaction length is sampled for each process using a different random number, proposing a *step*. The tracking scans all physics processes for the given particle and decides which one is to be invoked. The shortest interaction length determines the process to undertake unless a volume boundary of the experimental set-up is encountered in less than the sampled length. In that case no physics interaction will take place and the particle will be just transported to the geometry boundary, where a new step will take place.

The processes that were not chosen have their interaction lengths shortened by the distance travelled in the previous step Δx, according the formula:

$$n_\lambda^i = n_\lambda - \frac{\Delta x}{\lambda(x)}$$

Then the procedure is repeated.

The short description given above is the *differential approach* to particle transport, which is used in most simulation codes (see the Geant4 Physics Reference Manual [26]) In this approach besides the other (*discrete*) processes the continuous energy loss imposes a limit on the step size too because the cross sections depend on the energy of the particle. Then it is assumed that the step is small enough so that the particle cross sections remain approximately constant during the step. In principle one must use very small steps in order to insure an accurate simulation, but computing time increases as the step size decreases. A good compromise is to limit the step size in Geant4 by not allowing the stopping range of the particle to decrease by more than 20 % during the step (Chapter 5, Geant4 Physics Reference Manual [26]).

A Geant4 physics process has two methods: *GetPhysicalInteractionLength()* and *DoIt()*. The first one determines the interaction length of the specific process and the second one generates the final state (energy, momentum of particle, generation of secondaries, etc).

A Geant4 process can handle alternative or complementary physics models, as we will see more in detail in the next session. This strategy offers the possibility of providing multiple models for the same process. Models may be specialised, for example, for particle type, energy range, etc. The user can investigate and define which combination of physics models is the best to describe his/her particular case. This is done comparing the simulation results with experimental data and evaluating with quantitative comparison which physics approach provides the best agreement between the two data sets.

In Geant4 there are three 'pure' types of processes, as described in Table 1: well-located in space (*PostStep*), distributed in space (*AlongStep*), well-located in time (*AtRest*). A type of process can be pure *discrete*, *continuous*, or *atRest*, as indicated in Table 1 or can be a combination of different types. Examples of combination of continuous and discrete process are the ionisation and the bremsstrahlung interactions. In the case of ionisation, the energy loss is continuous, and the production of secondary is discrete; in the case of bremsstrahlung the energy loss due to soft photons is continuous and the hard photon emission is discrete. In both cases, the *production threshold* separates the continuous and the discrete parts of the process.

Table 1: "pure" types of Geant4 physics processes.

Type	Example
Discrete	Compton scattering
Continous	Cherenkov effect
AtRest	Positron annihilation

The threshold of production of secondary particles: the range cut

In Geant4, charged particles are tracked to the end of their range. However, for CPU performance, when generating the particles produced in an interaction, a process may, optionally, choose to suppress particles whose range would be less than a user-defined value, called *range cut* [1]. In this case the process must add the energy of the particle to the energy deposited during the step.

For some processes, such as δ-ray and bremsstrahlung production, the use of a cut is not an option but a necessity in order to suppress the generation of large numbers of soft electrons and gammas. The energy of non-produced particles is transferred from the discrete component of a process to the continuous (along-step) component. This production threshold concept is used by the electromagnetic processes, in particular by ionisation and bremsstrahlung [1].

One of the most relevant difficulties for a beginner of Geant4 is to understand how to fix the value of threshold of production of secondary particles. This depends strictly on the specific Geant4 study and a general rule cannot be defined. The best strategy to adopt, when determining the value of the range cut, is to compare the simulation results with experimental measurements, and reduce the value of the cut from the default one in order to achieve the best agreement between simulation and experimental data. To reduce the threshold costs in terms of CPU performance, as longer simulation execution time is required (more secondary particles are tracked in the experimental set-up). The optimised value of threshold is a compromise between accuracy of Geant4 simulation results and CPU performance.

Electromagnetic physics interactions

The Geant4 simulation Toolkit includes a number of packages to handle the electromagnetic interactions of electrons, muons, positrons, photons, hadrons and ions. Geant4 electromagnetic packages are specialised according to the particle type they manage, or the energy range of the process they cover.

The physics processes modelled in Geant4 electromagnetic packages include: multiple scattering, ionisation, bremsstrahlung, positron annihilation, photoelectric effect, Compton and Rayleigh scattering, pair production, synchrotron and transition radiation, Cherenkov effect, refraction, reflection, absorption, scintillation, fluorescence and Auger electron emission [1].

Alternative Geant4 electromagnetic models are provided by the Geant4 Standard and Low Energy electromagnetic packages [1].

The Geant4 Standard electromagnetic package [30] provides a variety of models based on an analytical approach, to describe the interactions of electrons, positrons, photons, charged hadrons and ions in the energy range 1 keV–10 PeV. The models assume that the atomic electrons are quasi free; their binding energy is neglected except for the photoelectric effect. The atomic nucleus is assumed to be fixed and its recoil momentum is neglected.

The Geant4 Low Energy electromagnetic package [31] extends the coverage of electromagnetic interactions in Geant4 below 1 keV, an energy range that is not covered by the Geant4 Standard package. It handles the interactions of electrons, positrons, photons, charged hadrons and ions, offering different sets of models for each of the physics processes involved. This package is particularly suited to low energy applications.

The interactions of electrons and photons are described by two alternative sets of models. The set of models based on a parameterised approach exploits evaluated data libraries (EPDL97 [32], EEDL [33] and EADL [34]). These data sets are used to calculate cross sections and to sample the final state.

Another set of models for electrons, positrons and photons is based on an approach combining numerical databases and analytical models for the different interaction mechanisms [35, 36]. These models were originally developed for the Penelope Monte Carlo FORTRAN code [37], and have been re-engineered into Geant4 with an object-oriented design.

Low energy processes are also available to handle the ionization by hadrons and ions [38], [39]. Different models, specialised for energy range, particle type and charge, are provided. Above 2 MeV, the Bethe–Bloch formula is applied; below 1 keV the interactions are described by the free electron gas model. In the intermediate energy range parameterised models based on experimental data from the Ziegler [40]–[42] and ICRU [43] reviews are implemented. Corrections due to the molecular structure of materials and the effect of the nuclear stopping power are also taken into account. The Barkas effect [44] is described by means of a specialised model.

The Geant4 Very Low Energy Package [45, 46] models particle interactions in water, the main substance of biological systems, down to the electronvolt scale. This physics component is of particular interest for nanodosimetric studies as [13].

Hadronic physics interactions

The basic requirements on the physics modelling of hadronic interactions in a simulation toolkit span more than ~15 orders of magnitude in energy. The energy ranges from thermal for neutron interactions, through 7 TeV (in the laboratory) for LHC experiments, to even higher for cosmic ray physics. In addition, depending on the experimental set-up to be simulated, the full range or only a small part might be needed in a single application.

The complex nature of hadronic showers and the particular needs of the experiment require the user to be able to vary easily the models for particular particles and materials, depending on the situation.

The Geant4 hadronic physics component provides description of hadronic elastic and inelastic scattering (0 - ~TeV), capture, fission, radioactive decay (at-rest and in flight), photo-nuclear (~10 MeV - ~TeV), and lepto-nuclear reactions (~10 MeV - ~TeV).

The Geant4 hadronic package addresses the intrinsic complexity of this physics domain through a sophisticated software design [1]. The design identifies the process involved (i.e. elastic or inelastic scattering) and defines a framework for the articulation of the different models implementing them. A hadronic process is implemented by one or more (1) cross section (determining the mean free path) (2) models (determining the final state).

Models are characterised by different conceptual approaches (i.e. parameterised or theory-driven), by the energy range covered and by specific theoretical approach adopted (i.e. Binary or Bertini cascade for intranuclear transport).

For detail description of the various Geant4 models please refer to the Geant4 Physics Reference Manual, downloadable from [26]. Here we will introduce some of the most used hadronic models, adopted to describe the

interactions of hadrons up to ~GeV scale, of interest for radiotherapy, radiation protection and development of detectors to be used in radiation fields with energy lower than few GeV.

Below ~3 GeV centre of mass energy, intra-nuclear transport models are provided. For cascade type models, the alternative Bertini [48] and Binary [49] cascade models are provided, to describe the inelastic scattering interactions of neutrons, protons, pions, kaons, and hyperons.

At energies below O(100 MeV), Geant4 provides the possibility of using exciton based pre-compound models (Geant4 Physics Reference Manual)[26] to describe the energy and angular distributions of the fast particles in the pre-equilibrium phase. The precompound stage of nuclear reaction is considered until nuclear system is not an equilibrium state. Further emission of nuclear fragments or photons from excited nucleus is simulated using an equilibrium model (nuclear de-excitation and evaporation models).

The default Geant4 nuclear evaporation model considers the emission of light fragments only (up to alpha particles). In alternative it is possible to select the Generalised Evaporation Model (GEM) by Furihata (Geant4 Physics Reference Manual)[26]. This model considers also the possibility of emission of fragments heavier than alpha particles.

The Binary cascade activates internally the Geant4 pre-compound model, while the Bertini cascade embeds its own pre-compound and de-excitation model.

The Geant4 hadronic physics package includes also the Binary Ion Model to describe the interactions of ions, of incident energy lower than 10 GeV/nucl [50].

The low energy neutron HP library (Chapter 35, Physics Reference Manual [26]) is specialised to describe the interactions of neutrons with kinetic energies from thermal energies up to O(20 MeV). Its adoption is highly suggested when accurate modelling of neutron interactions is required at low energy.

MICROSCOPIC AND MACROSCOPIC VALIDATION OF GEANT4

When we develop a Geant4 simulation, e.g. to verify the accuracy of a treatment planning, or to improve a radiotherapy treatment, we have to be certain the software tool we are using is reliable.

The method to quantify the accuracy of a Geant4 simulation application (as for any other Monte Carlo code) is to validate it with respect to a reference. The validation must be performed with respect to in-house or external experimental data, or any published recognised data. The comparison of Geant4 with respect to other Monte Carlo codes is not a validation activity, but a simple comparison of mathematical algorithms.

There are mainly two activities to quantify the accuracy of a Monte Carlo code: the microscopic and macroscopic validation of the software tool.

In the first case the object of the validation is the reliability of the physics models, independent from any specific application. The validation of the physics models should be done in a simple experimental set-up, so as to obtain results dependent on the physics models only, and not on the experimental configuration.

In the macroscopic validation, the reliability of the specific Geant4-based simulation is tested, considering not only the physics models, but also the modeling of the geometrical experimental set-up, of the response of detectors, etc. The macroscopic validation is highly dependent on the specific use case.

The difficulty of a validation study (as for any other Monte Carlo code) is determined mainly by the following reasons:

(1) Reliable reference data must be individuated, as a reference; these should be obtained in a highly controlled experiment, where all the parameters of the experiment are known precisely. Often published experimental measurements do not provide details on the experimental set-up and cannot be used for validation. In this context, the use of in-house reference data simplifies the validation activity, as the experimental configuration is under control and well known in most of these cases.

(2) The Geant4 simulation set-up should reproduce in detail the experimental set-up of the reference data. This factor determines the validity of the testing process. One of the most typical errors in this context is to compare simulation results and experimental data, with different binning. It is well known that we achieve better agreement between simulation results and experimental measurements when a more accurate binning is adopted in the Geant4 simulation.

(3) The comparison must be quantitative, by means of statistical methods.

Validation activity is very fruitful as it significantly supports the improvement of the physics models offered by Geant4. There are many scientific papers published in international journals concerning microscopic validation studies of Geant4. We will just list some publications that the reader may start with. For example [51] presents the

results of comparisons of Geant4 electromagnetic processes of photons, electrons, protons and alpha particles with respect to reference data of the United States National Institute of Standards and Technologies (NIST) [52], [53] and of the International Commission on Radiation Units and Measurements (ICRU) [54], [55]. More recent validation studies of Geant4 electromagnetic physics are offered by [56], [57], [58]. Reference [29] is a very good guide to understand which Geant4 physics models are more adequate to describe proton fields, of interest in protontherapy and radiation protection. Moreover this paper describes in detail the procedure we retain more successful to individuate the best Geant4 physics list describing a specific use case.

A GEANT4 SIMULATION APPLICATION

Geant4 is a simulation Toolkit, so it is not possible to "run" it out the box, but the user has to develop a Geant4 application, providing all the necessary information to configure the simulation and selecting the Geant4 tools to use.

In particular the user has to describe in his/her Geant4 application the experimental set-up, in terms of geometry and materials and the radiation field, input to the simulation. The user has also to decide which particles and physics models to instantiate in the simulation, the threshold of production of secondary particle and eventual other parameters (i.e. the maximum step allowed in regions of the experimental set-up). The user may want also to interact with Geant4 kernel to control the simulation, to visualise the experimental set-up and particle track, produce histograms, ntuples, etc., containing the results of the simulation to be further analysed.

A Geant4 application consists in the creation of concrete classes, derived from the base user classes of Geant4, that are tools used to communicate to the kernel about the simulation configuration and to interact with the Geant4 kernel itself. The base user classes are listed in Table 2. *G4VUserDetectorConstruction*, *G4VUserPhysicsList* and *G4VPrimaryGeneratorAction* are abstract and the user derived concrete classes are mandatory. The other base user classes are concrete with virtual dummy methods. In this case the user derived classes are optional.

The *G4VUserDetectorConstruction* manages the definition of the experimental set-up in terms of geometry and material composition. The *G4VUserPhysicsList* manages the instantiation of the particles involved in the experimental set-up, of their interactions in matter, of the definition of the threshold of production of secondary particles. The *G4VUserPrimaryGeneratorAction* handles the model of the radiation field, input of the simulation.

The *G4UserRunAction* allows to gain control of the simulation at the beginning and at the end of the run. For example the user may want to use this class to retrieve the total number of events of the run, or the total energy deposition in a detector, at the end of the run.

TABLE2. User base classes of Geant4

Base User Class Type	Base User Class	User Derived Class
Initialisation	G4VUserDetectorConstruction	Mandatory
	G4VUserPhysicsList	Mandatory
Action	G4VUserPrimaryGeneratorAction	Mandatory
	G4UserRunAction	Optional
	G4UserTrackingAction	Optional
	G4UserStackingAction	Optional
	G4UserSteppingAction	Optional

The *G4UserTrackingAction* and the *G4UserEventAction* allow to gain control of the simulation at the beginning and at the end of each track processing and event, respectively.

The *G4UserTrackingAction* may be used to retrieve information of the tracks of particles in the experimental set-up. For example the type, energy and vertex of generation of secondary particles, etc., can all be accessed.

The *G4UserStackingAction* allows customisation of the stacking mechanism of the particle tracks. For example the user may use this class to kill the tracks of particles in specific regions of the experimental set-up.

The *G4UserSteppingAction* allows retrieval of information at the level of the step. The user may retrieve at this level the energy loss of a particle during the step, the kinetic energy and momentum of particles, etc.

The software architecture of a typical and basic Geant4 application is depicted in Fig. 5, using the Unified Modeling Language (UML) [59]. The user has to derive his/her own concrete classes from the Geant4 base classes. The user concrete classes are the Geant4 simulation application. The reader may refer to [59] to understand the meaning of the arrows in Fig. 5, indicating the type of relation among the classes of the Geant4 application and the Geant4 kernel.

Geant4 does not provide the *main()* method. The user has to write his/her own *main()* method, where the user has to instantiate the *G4RunManager* and three mandatory concrete classes (MyDetectorConstruction, MyPhysicsList, and MyPrimaryGeneratorAction in Fig. 5), inheriting from *G4VUserDetectorConstruction, G4VUserPhysicsList* and *G4VUserPrimaryGeneratorAction*. The user can define a Visualization Manager, (G)UI session, optional user action classes in the *main(),* to customize the behaviour of the Geant4 application.

As guidance, the student may refer to the Geant4 novice and advanced examples, released within the Toolkit software, or to the `basic_brachy` application provided within this course, to retrieve the concepts explained in this session and to adopt as a start to develop a new Geant4 simulation application.

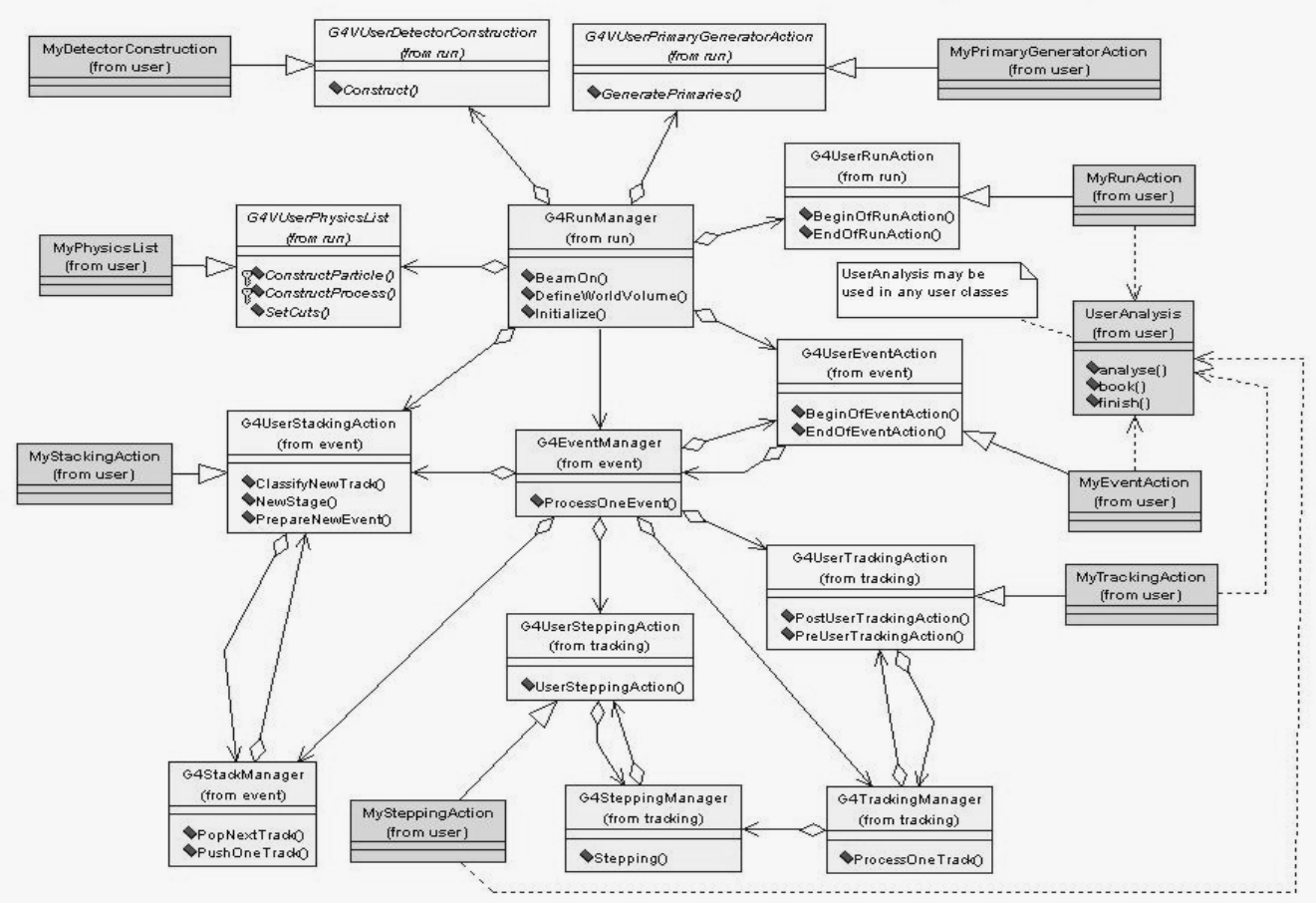

Figure 5: Design of a typical Geant4 application, using the Unified Modeling Language (UML) [59]. Picture courtesy of Geant4 Collaboration.

The Geant4 simulation: Initialisation and Beam-On phases

Object of this session is the dynamic flow of the Geant4 simulation, during its execution. A Geant4 simulation has two major phases: initialisation and beam-on.

The initialisation (represented in Fig. 6) is triggered by the *Initialize()* method of *G4RunManager*; the experimental set-up (in terms of geometry and material composition) of the simulation is initialised, followed by the physics processes of all the particles involved in the experimental set-up. Then the beam-on phase starts triggered by the *BeamOn()* method of the *G4RunManager*. The geometrical set-up is closed and optimised and then the event loop follows, as illustrated in Fig. 7. The events are generated as shown in Fig. 8, the primary particle and all the

secondary particles are tracked in the experimental set-up. Then the event is closed and a new one is generated. The simulation ends when all the events of the *BeamOn* have been processed.

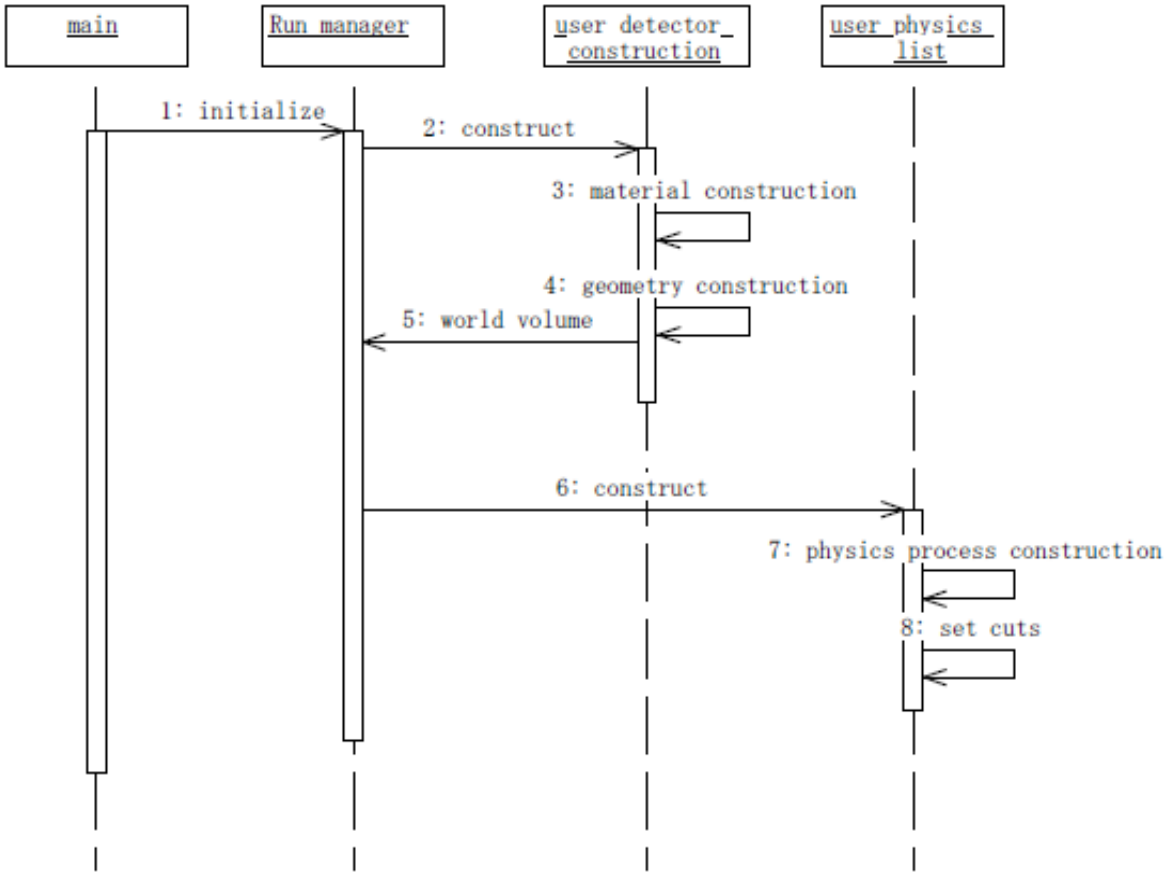

Figure 6: Initialisation phase of a Geant4 simulation [60]. Picture from [60].

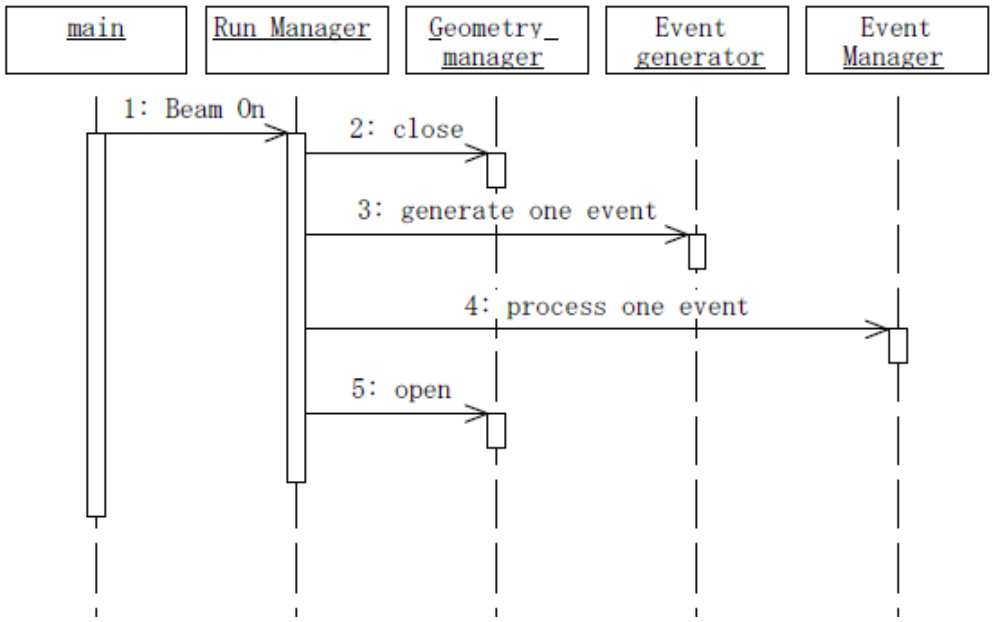

Figure 7: Run phase of a Geant4 simulation [60]. Picture from [60].

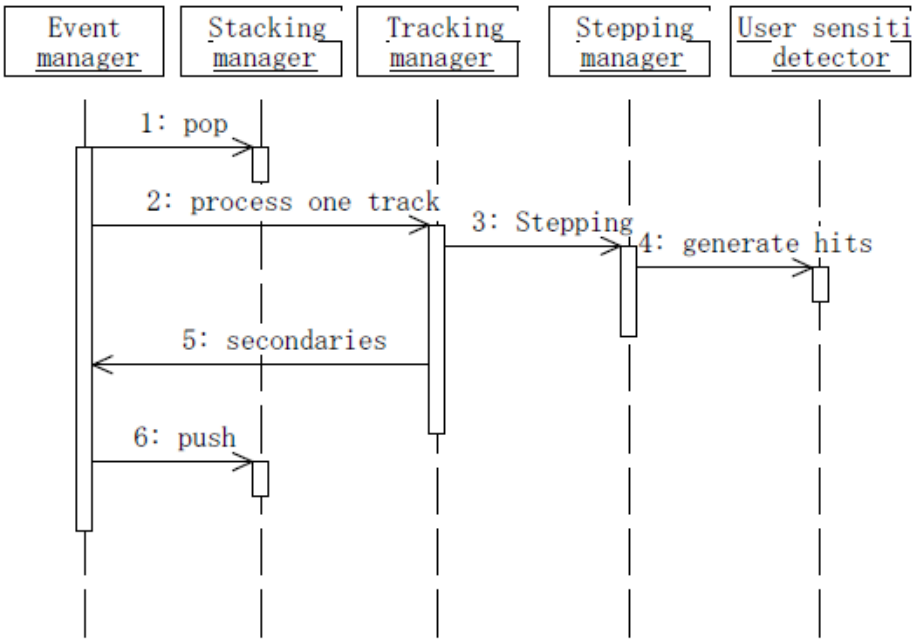

Figure 8: Diagram showing the processing of one event of the Geant4 simulation [60]. Picture from [60].

HOW TO DEVELOP A GEANT4 APPLICATION

Software process

The best approach, when starting from scratch to develop a Geant4 application is to get a clear vision of the project which is to be developed. The student should completely understand the goal and the motivation of his/her

project. Once the student understands in depth his/her project, then must obtain a clear idea of the goal of the Geant4-based study.

The second step involves understanding in detail what the Geant4 simulation application should do. This means that the user should know in detail the experimental set-up to model in the Geant4 application, in terms of (1) geometry and material composition, (2) characteristics of the radiation field in terms of particle type, energy, spatial and direction distributions, (3) the interactions of all the particles involved in the experimental set-up, (4) the physics quantities to retrieve as output of the simulation.

The third step is to download the *Geant4 Application Developer Manual* [26], which should be adopted as a reference manual. This guide is very helpful, especially for Geant4 beginners, as it explains how to develop a Geant4 application.

A fundamental method to adopt in the development of a software tool is the software process, encompassing the set of activities, methods, practices, and transformations that people use to develop and to maintain software and associated products [61]. The software process is a key element to ensure that a project matches its defined objectives. Its adoption is particularly important when developing projects, involving more developers, in the same time frame, or in successive phases on project evolution. At the same time it enhances the efficiency of the work [61]. Any interested student may start to read bibliographic references from [62] to [65] on this subject.

The student may learn to develop a Geant4 application, from the `basic_brachy` application provided, through the series of exercises suggested at the end of the paper.

The basic_brachy application

A very common practice, that we find successful, is to start to develop a Geant4 application, adopting a Geant4 application example. As support, we provided a very simple Geant4 application, the `basic_brachy`, which the students may want to adopt to start to develop their own application. The student should thoroughly understand this application, in terms of functionality and implementation details. After that, the Geant4 beginner can start to modify the application in order to satisfy his/her application requirements, adopting an interactive, incremental method and the software development process [62].

We suggest using Linux as platform, in particular the CERN Scientific Linux distribution, as this is the Geant4 development platform (http://linux.web.cern.ch/linux/). To learn basic commands of Linux refer to [66].

In this course, we give for granted that Geant4 is already installed on your computer. If you do not know how to install the software, you may want to consult the webpage http://geant4.slac.stanford.edu/installation/. You should also install the analysis tool ROOT [67], to activate the analysis component of the `basic_brachy` application. You may also decide not to use ROOT and to store the results of the simulation in an ASCII file or you may also decide to use an alternative analysis tool, to interface to your Geant4 simulation application.

The `basic_brachy` application is a dosimetric system. Its architecture is typical of a Geant4 application (see Fig. 5). Fig. 9 shows a sketch of the experimental set-up. A point-shaped, isotropic source of photons, with energy equal to 300 keV, is set in the center of a water phantom (limited by the grey line in Fig. 9), with size equal to 30 cm, modelling a typical phantom used for dosimetry, in medical physics studies. The Geant4 Low Energy Package [31] has been selected to describe the interactions of electrons and photons, particles involved in the experimental set-up of the application. The processes are: Rayleigh scattering, photoelectric effect and Compton scattering for photons; ionization, bremsstrahlung and multiple scattering for electrons. The threshold of production of secondary particles (cut) is fixed equal to 1 mm. The result of the simulation is the energy deposition in the water phantom (defined as sensitive volume), deriving from the secondary electrons produced by photon interactions.

The output of the simulation is a ROOT file (`brachytherapy.root`), containing a ntuple with all the energy depositions (in keV) in the phantom, with respect to the position x, y, z, expressed in mm. The user can use a macro, `macro.C`, to use in the interactive session of ROOT. This helps to visualize and analyse the results, as shown in Fig. 10.

The application `basic_brachy` is provided with a simple user interface, to define parameters of the experimental set-up (i.e. number of photons to shoot as primary particles, energy of the emitted photons, etc.) in the simulation interactive sessions, or in macro files to be executed in batch mode.

The `basic_brachy` directory contains the Geant4 application. The directory contains a main, named `basic_brachy.cc`, the macro files `run.mac` (to run the application in batch mode), `vis.mac` (to activate the visualisation of the experimental set-up and particle tracks during the execution of the simulation) and `macro.C` (for ROOT data analysis). The directories `include` and `source` contain the header and cc files, implementing the classes of the application. The classes of the application are listed in Table 3.

TABLE3. Classes of the `basic brachy` Geant4 application.

Classes of `basic brachy`	Responsibility
`AnalysisManager`	Creating, filling and closing the output ROOT file, containing the results of the simulation.
`DetectorConstruction`	Model of the experimental set-up in terms of geometry and material composition.
`PhysicsList`	Instantiation of particles involved in the experimental set-up and their physics interactions. Definition of the threshold of production of secondary particles.
`PrimaryGeneratorAction`	Model of the radiation field, input of the simulation.
`RuAction`	At the end of the run, the number of events is printed out on the screen.
`SensitiveDetector`	Instantiation of the water phantom as sensitive volume, where the energy deposition is retrieved.

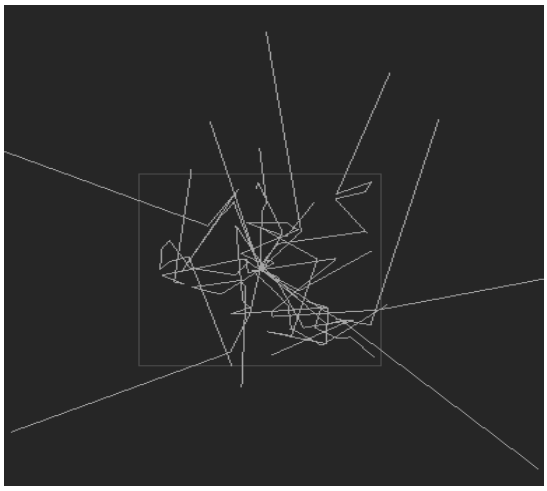

Figure 9: Experimental set-up of the `basic_brachy` application.

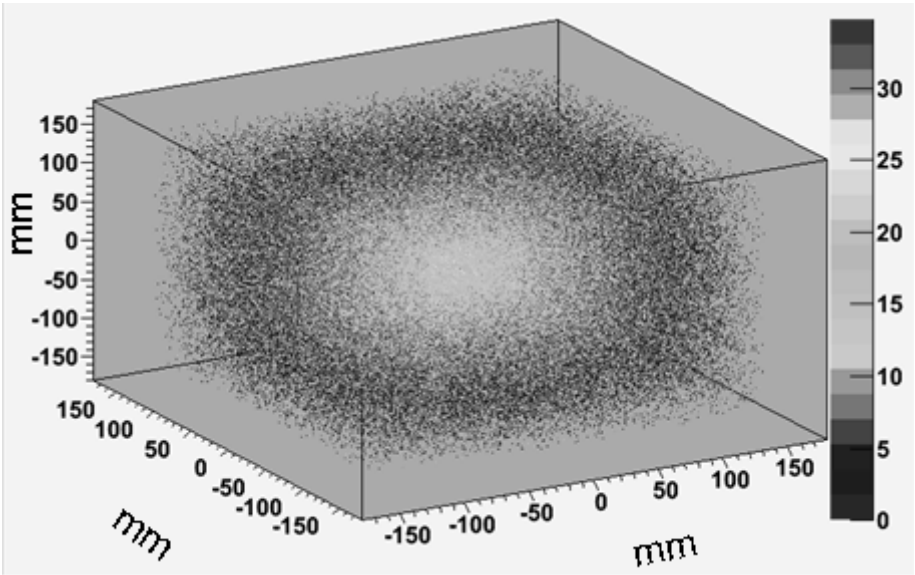

Figure 10: Energy deposition in the phantom, in 3D representation. Each grey tone represents a defined energy deposition, as indicated on the right scale. The energy deposition is in keV; the position x, y, z is in mm. This result was obtained, with a simulation consisting of 1000 events.

How to execute the basic_brachy application

The student should compile the `basic_brachy` with the command `gmake` and type `$G4WORKDIR/bin/Linux-g++/basic_brachy` to execute the simulation in interactive mode. To run the simulation in batch mode, type: `G4WORKDIR/bin/Linux-g++/basic_brachy run.mac`.

GEANT4 EXERCISES

The following set of exercises is a guide for students to develop a dosimetric system for brachytherapy, starting from the `basic_brachy` application. The suggestion is to adopt an iterative-incremental approach, as explained in the "Software process" section. Remember to verify the correct functionality of the software, after each development cycle. For questions contact Susanna Guatelli (e-mail: susanna@uow.edu.au).

Exercise 1

Run the `basic_brachy` simulation in interactive and batch mode. Analyze the results contained in the file `brachytherapy.root`, executing the macro `macro.C` in an interactive ROOT session.

Exercise 2

Model a brachytherapic source (Fig. 11), in the experimental set-up of the Geant4 simulation, in term of geometry and material composition.

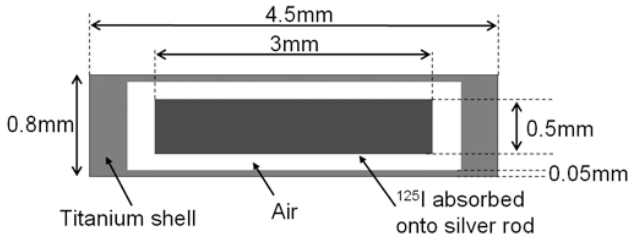

Figure 11: Sketch of the brachytherapy ^{125}I source, model 6711, by Oncura [68]**.**

Exercise 3

Model the radiation field of the brachytherapic source depicted in Fig. 11.

Exercise 4

Calculate the dose distribution in the plane containing the radioactive source, with cut equal to 1 μm and 1mm. What do you observe?

Exercise 5

Change the electromagnetic physics models from Livermore data libraries based to Penelope approach.

Exercise 6

How can you validate your Geant4 simulation application?

ACKNOWLEDGMENTS

We would like to thank Makoto Asai, Emma Simpson, Jayde Livingstone and Stephen Dowdell, for providing precious feedback to this paper.

The authors appreciate the Geant4 Collaboration for its permission of the use of figures 1, 3, 4, 5, 6, 7 and 8.

The authors appreciate the Geant4 Collaboration for its permission of the use of figures and texts from their publications, users guide and tutorial materials.

The `basic_brachy` application was developed for the PHYS366/PHYS952/PHYS950 course of UOW [69].

REFERENCES

1. Geant4 Collaboration (S. Agostinelli et al.), Nucl. Instrum. Meth. Phys. Res. A **506**, 250-303 (2003).
2. Geant4 Collaboration (J. Allison et al.), IEEE Trans. Nucl. Sci., **53**, 270-278 (2006).
3. A. B. Rosenfeld et al, IEEE Trans. Nucl. Sci., **47**, 1386–1394 (2000).
5. I. M. Cornelius, et al, IEEE Trans. Nucl. Sci., **50,** 2373-2379 (2003).
6. A. Wroe et al, IEEE Trans. Nucl. Sci., **52**, 2591 – 2596 (2005).
7. A. Rosenfeld et al, Rad. Prot. Dosim., **119**, 487–490 (2006).
8. A. Wroe et al, IEEE Trans. Nucl. Sci., **53,** 3738-3744 (2006).
9. D. A. Prokopovich et al, Rad. Meas., **43**, 1054–1058 (2008).
10. S. Guatelli et al, IEEE Trans. Nucl. Sci., **55**, 3407-3413(2008).
11. M. A. R. Othman et al, Rad. Meas., accepted on 22 Jun 2010.
12. M. A.R Othman et al, Rad. Prot. Dos., **141**, 10-17(2010).
13. M.U. Bug et al, European Physical Journal D, 2010, DOI: 10.1140/epjd/e2010-00145-1, Published online: 26 May 2010.
14. S. Dowdell, et al, Med. Phys., **36**, 5412-5419 (2009).
15. N. Metropolis. The beginning of the monte carlo method. Los Alamos Science Special Issue, 125-130, (1987).
16. Lazzarini's Lucky Approximation of π, Mathematics Magazine (Mathematical Association of America) 67, 83-91.
17. W. R. Nelson, H. Hirayama, D. W. O. Rogers, "The EGS4 code system," SLAC-256. Stanford Linear Accelerator Center, Stanford, California (1985).
18. F.Salvat, et al, "PENELOPE, A Code System for Monte Carlo Simulation of Electron and Photon Transport", Proceedings of a Workshop/Training Course, OECD/NEA 5-7 November 2001.
19. MCNPX User's Manual, Version 2.3.0, L. S. Waters, Ed., LANL Report LA-UR-02-2607, 2002. Webpage:http://mcnpx.lanl.gov/.
20. S. Giani, et al, CERN Document CERN/LHCC 98-44 (1998).
21. Website: http://geant4.esa.int/
22. L. Archambault, et al, *Proc. Conf. Rec. 2003 IEEE Nucl. Sci. Symp.*, Portland, USA (2003).
23. Website: http://geant4.cern.ch/applications/hepapp.shtml
24. S. Chauvie, et al, IEEE Trans. Nucl. Sci., **54**, 2619-2628 (2007).
25. S. Incerti et al, Med. Phys., **37**, 4692-4708 (2010).
26. Manuals available at the website: http://geant4.web.cern.ch/geant4/support/userdocuments.shtml
27. Website: geant4.kek.jp/~tanaka/**DAWN**/
28. Website: http://reat.space.qinetiq.com/gps/new_gps_sum_files/gps_sum.htm
29. C. Zacharatou Jarlskog et al. , IEEE Trans. Nucl. Sci., **55**, 1018-1025 (2008).
30. V. N. Ivanchenko, et al, "Geant4 standard electromagnetic package for HEP applications," in Proc. Conf. Rec. 2004 IEEE Nucl. Sci. Symp., Rome, Italy (2004).
31. S. Chauvie, et al, "Geant4 low energy electromagnetic physics", in Conf. Rec. 2004 IEEE Nuclear Science Symp., Rome, Italy (2004).
32. D. Cullen, et al, Lawrence Livermore National Laboratory, Rep. UCRL-50 400, **6**, 1997.
33. S. T. Perkins, et al, Lawrence Livermore National Laboratory, Rep. UCRL-50 400, **31** (1997).
34. S. T. Perkins, et al, Lawrence Livermore National Laboratory, Rep. UCRL-50 400, **30** (1991).

35. J. Sempau, et al, Nucl. Instrum. Methods Phys. Res. B **132**, 377–390 (1997).
36. J. Baro, et al, Nucl. Instrum. Methods Phys. Res. B, **100**, 31–46 (1995).
37. J. Sempau, et al, "PENELOPE—a code system for Monte Carlo simulation of electron and photon transport," in Proc. Workshop Issy les Moulineaux, Issy les Molineaux, France, 1–253, 2001.
38. S. Giani, et al, INFN, Tech. Rep. INFN/AE-99/20 (1999).
39. S. Giani et al, Tech. Rep. INFN/AE-99/21 (1999).
40. H. H. Andersen and J. F. Ziegler, *The Stopping and Ranges of Ions in Matter*. New York: Pergamon, 1977, vol. 3.
41. J. F. Ziegler, J. P. Biersack, and U. Littmark, *The Stopping and Ranges of Ions in Solids*. New York: Pergamon, 1985, vol. 1.
42. J. F. Ziegler and J. P. Ziegler, *The stopping and range of ions in matter*, IBM-Research, Yorktown, NY, 2000.
43. ICRU Rep. 49, Bethesda, MD (1993).
44. S. Chauvie et al, IEEE Trans. Nucl. Sci., **54**, 578-584 (2007).
45. S. Chauvie et al, IEEE Trans. Nucl. Sci, **54**, 2619-2628 (2007).
46. S. Incerti et al, Med. Phys., 37, 4692-4708 (2010).
47. M. Bug et al, Europ. Phys. Journ. D, 2010, DOI: 10.1140/epjd/e2010-00145-1, Published online: 26 May 2010.
48. A. Heikkinen, et al, Proc. of CHEP 2003, Int. Conf. On Computing in High Energy and Nuclear Physics, paper MOMT008 (2003).
49. G. Folger et al, Proc. of CHEP 2000, Int. Conf. On Computing in High Energy and Nuclear Physics, paper 256 (2004).
50. T. Koi et al, Proc. of CHEP 2004 Int. Conf. On Computing in High Energy and Nuclear Physics, paper 255 (2004).
51. K. Amako, et al, IEEE Trans. Nucl. Sci., **52**, 910-918 (2005).
52. M. J. Berger, et al, *XCOM: Photon Cross Section Database (Version 1.2)*. Gaithersburg, MD: National Institute of Standards and Technology, 1999.
53. M. J. Berger, et al, *ESTAR, PSTAR, and ASTAR: Computer Programs for Calculating Stopping-Power and Range Tables for Electrons, Protons, and Helium Ions (Version 1.2.2)*. Gaithersburg, MD: National Institute of Standards and Technology, 2000.
54. "Stopping powers for electrons and positrons", ICRU Rep. 37, Bethesda, MD, 1984.
55. "Stopping powers and ranges for protons and alpha particles," ICRU Rep. 49, Bethesda, MD, 1993.
56. B. A. Faddegon et. Al, Med. Phys. **35**, 4308-4317 (2008).
57. A. Lechner et al, IEEE Trans. Nucl. Sci., **56**, 398-416 (2009).
58. S. Guatelli, et al, IEEE Trans. Nucl. Sci., **54**, 594-603 (2007).
59. M. Fowler, *UML Distilled, Third edition, a brief guide to the Standard Object Modeling Language,* Ed. Addison-Wesley.
60. M. Asai, Introduction to Geant4, http:cdsweb.cern.ch/record/491492/files/p107.ps.gz
61. S. Guatelli et al, "Experience with the Unified Process in simulation projects", The Monte Carlo Method: Versatility Unbounded in a Dynamic Computing World, Chattanooga, Tennessee, April 17-21, 2005, on CD-ROM, American Nuclear Society, LaGrange Park, IL, 2005.
62. P. Kruchten, *The Rational Unified Process, An Introduction*, Ed. Addison Wesley Longman, 2000.
63. G. Booch, J. Rumbaugh, and I. Jacobson, *UML User Guide*., Ed. Addison-Wesley Longman, 1998.
64. A. Kuntzmann–Combelles, P. Kruchten, *The Rational Unified Process – An Enabler for Higher Process Maturity,* http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/rup_tp178.pdf
65. G. Cosmo, "Geant4 software process," in Proc. CHEP Conf., Bejing, China, Sept. 2001.
66. E. Siever, S. Figgins, R. Love, and A. Robbins, *Linux in a nut shell*, Ed. O'Reilly Media, 6th Ed, 2009.
67. Web page: http://root.cern.ch/drupal/
68. Web page: http://www.oncura.net/
69. S. Guatelli et al, "Transferring advanced physics research tools to education: how to teach simulation tools used in radiation physics research to university students", Proceeding Records, International Technology, Education and Development Conference (INTED2010), Valencia, Spain, 8-10 March 2010.